WHAT IS CLAIMED IS:

1. A method for computing distances between a received point and four points in a two-dimensional grid with a constellation representing a number of bits greater than three, wherein each of the four points belong to a unique coset in the constellation, the method comprising:

determining a first point on a grid nearest to the received point;

computing a second point closest to the received point inside a specified area;

computing a third, fourth, and fifth point, wherein each point is a member of a different coset and each point is the closest point in its coset to the received point; and

computing a distance from the received point to each of the second, third, fourth, and fifth points.

2. The method of claim 1 further comprising after the first computing, recomputing the second point if the second point is invalid.

3. The method of claim 2, wherein the second point is invalid if it is outside of the constellation.

4. The method of claim 1, wherein the first point can be determined by evaluating:

$$\mathrm{round}((Rx + iRy - 1 - i)/2*2 + 1 + I,$$

wherein Rx and Ry are two-dimensional components of the received point, i is the imaginary number, and round(.) is an operator that returns an integer number closest to a value provided to it.

5. The method of claim 1, wherein the number of bits is an even value, wherein the received point can be expressed in two-dimensional components Rx and Ry, and wherein the first

computing comprises:

determining if Rx and Ry lie inside a square specified by the number of bits; and

computing two-dimensional components of the second point based on the second determining.

6. The method of claim 5, wherein the second determining comprises:

setting Cx = 1 if Rx lies inside a boundary of the square, else Cx = -1;

setting Cy = 1 if Ry lies inside a boundary of the square, else Cy = -1;

and wherein the fourth computing comprises

setting Ax = sign(RGx) * MAX_{XY} if Cx = -1, else Ax = RGx; and

setting Ay = sign(RGy) * MAX_{XY} if Cy = -1, else Ay = RGy,

wherein Ax and Ay are two-dimensional components of the second point, RGx and RGy are two-dimensional components of the first point, MAX_{XY} is value describing the size of the square and can be computed by $2^{\text{number of bits/2}} - 1$.

7. The method of claim 5, wherein the second computing comprises:

computing an intermediate value, d, wherein d = the received point – the second point;

setting the third point = the second point + Cx * sign(dx) * 2;

setting the fourth point = the second point + i * Cy * sign(dy) * 2; and

setting the fifth point = the second point + 2(Cx * sign(dx) + i * Cy * sign(dy)),

wherein Cx and Cy are values specifying if the two-dimensional components of the received point lie inside a boundary of the square and dx and dy are two-dimensional components of d.

8. The method of claim 5, wherein the third computing comprises computing a Euclidean distance from the received point to each of the second, third, fourth, and fifth points.

9. The method of claim 8, wherein each of the second, third, fourth, and fifth points belong to a unique coset.

10. The method of claim 1, wherein the number of bits is an odd value greater than three, wherein the received point can be expressed in two-dimensional components Rx and Ry, and wherein the first computing comprises computing two-dimensional components of the second point, wherein the second point lies within a first square.

11. The method of claim 10, wherein the first square encompasses a cross-shaped constellation, wherein a value MAX_{XY} describes the size of the first square and can be computed by expression $(2^{\lceil \text{number of bits}/2 \rceil} + 2^{\lfloor \text{number of bits}/2 \rfloor})/2-1$, wherein a second square that is a largest square encompassing 2^N constellation points with N being an even integer that can be enclosed in the first square, wherein MAX_X and MAX_Y describe the size of the second square, and wherein the first computing comprises:

setting Ax = sign(RGx) * MAX_{XY} if abs(RGx) > MAX_{XY} else Ax = RGx; and

setting Ay = sign(RGy) * MAX_{XY} if abs(RGy) > MAX_{XY} else Ay = RGy,

wherein Ax and Ay are two-dimensional components of the second point, and RGx and RGy are two-dimensional components of the first point.

12. The method of claim 10 further comprising after the first computing, recomputing the second point if the second point is invalid.

13. The method of claim 12, wherein the second point is invalid if (abs(Rx) > MAX_X and abs(Ry) > MAX_Y) is true.

14. The method of claim 12, wherein the recomputing comprises

setting the second point = Ax + i * sign(Ay) * MAX_Y

and a sixth point = sign(Ax) * MAX_X + i * Ay if abs(Rx) > abs(Ry)

else setting the second point = sign(Ax) * MAX_X + i * Ay

and the sixth point = Ax + i * sign(Ay) * MAX_Y,

wherein Ax and Ay are two-dimensional components of the invalid second point and α is another point in the two-dimensional grid.

15. The method of claim 14, wherein the second computing comprises:

determining if the second point lies on the boundaries of the first square and α lies on the boundaries of a second square, wherein the second square is the largest square encompassing 2^N constellation points with N being an even integer that can be enclosed in the first square, wherein MAX_X and MAX_Y describe the size of the second square; and

computing the third, fourth, fifth points.

16. The method of claim 15, wherein the second determining comprises:

setting Cx = -1 if (((abs(Rx) >= MAX_X) and (abs(Ay) > MAX_Y)) or (abs(Rx) >= MAX_{XY})) is true, else Cx = 1;

setting Cy = -1 if (((abs(Ry) >= MAX_y) and (abs(Ax) > MAX_X)) or (abs(Ry) >= MAX_{XY})) is true, else Cy = 1;

setting C'x = 1 if R'x lies within the boundaries of the second square else C'x = -1; and

setting C'y = 1 if R'x lies within the boundaries of the second square else C'y = -1,

wherein R'x and R'y are the two-dimensional components of α.

17. The method of claim 15, wherein the second computing comprises:

computing an intermediate value, d, wherein d = the received point – the second point;

setting the third point = the second point + Cx * sign(dx) * 2;

setting the fourth point = the second point + i * Cy * sign(dy) * 2; and

setting the fifth point = the second point + 2(Cx * sign(dx) + i * Cy * sign(dy)),

wherein Cx and Cy are values specifying if the two-dimensional components of the received point lie inside a boundary of the constellation and dx and dy are two-dimensional components of d.

18. The method of claim 17, wherein the second computing further comprises:

setting a seventh point = the sixth point + C'x * sign(d'x) * 2; and

setting an eighth point = the sixth point + i * C'y * sign(d'y) * 2,

wherein d'x and d'y are the two-dimensional values of a second intermediate value, d', wherein d' = the received point – the sixth point, and C'x and C'y are values specifying if the two dimensional components of the sixth point lie inside a boundary of the second square.

19. The method of claim 18, wherein the second computing further comprises checking to ensure that the fifth point is a valid point.

20. The method of claim 19, wherein the validity of the fifth point can be checked and corrected using expression:

if (abs(REAL(D)) > MAX_X) and (abs(IMAG(D)) > MAX_Y))

{ if (abs(dx) > abs(dy)) then

D = D – 4 * sign(IMAG(D)) * i,

else

D = D – 4 * sign(REAL(D)) },

wherein D is the fifth point, and dx and dy are the two-dimensional components of the fifth point.

21. The method of claim 18, wherein the third computing comprises computing a Euclidean distance from the received point to each of the second, third, fourth, fifth, sixth, seventh, and eighth points.

22. The method of claim 21, wherein each of the second, third, fourth, fifth, sixth, seventh, and eighth points belongs to one of four unique cosets, and wherein in a coset with more than one point, points with larger Euclidean distances to the received point are discarded.

23. The method of claim 1, wherein the method can be used to decode a received point in a communications system.

24. The method of claim 23, wherein the communications system is an asymmetric digital subscriber line (ADSL) compliant system.

25. A method for computing distances between a received point and four points in a two-dimensional grid with a constellation representing a number of bits equal to three, wherein each of the four points belong to a unique coset in the constellation, the method comprising:

computing a point from each coset to the received point; and

computing a distance from the received point to the point in each coset.

26. The method of claim 25, wherein the first computing comprises:

setting a first point = $1 + i$ if $Rx > -1$, else the first point = $-3 + i$;

setting a second point = $-1 - i$ if $Rx < 1$, else the second point = $3 - i$;

setting a third point = $1 - i$ if $Ry < 1$, else the third point = $1 + 3i$; and

setting a fourth point = $-1 + i$ if $Ry > -1$, else the fourth point = $-1 - 3i$;

wherein Rx and Ry are the two component values of the received point.

27. The method of claim 25, wherein the second computing comprises computing a Euclidean distance between the received point and the point in each coset.